#16

Attorney Docket No. UK999-027 303

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent Application

Applicant(s): H. E. Butterworth et al.
Docket No.: UK999-027
Serial No.: 09/401,676
Filing Date: September 22, 1999
Group: 2131
Examiner: Christian A. La Forgia

Title: Data Processing Systems and Method for Processing Work Items in Such Systems

I hereby certify that this paper is being deposited on this date with the U.S. Postal Service as first class mail addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Signature: V Brackenn Date: May 3, 2004

SUPPLEMENTAL APPEAL BRIEF

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313

Sir:

This Supplemental Appeal Brief is submitted in response to the Office Action dated February 2, 2004 in the above-referenced application, in which the Examiner reopened prosecution in response to the Appeal Brief filed September 8, 2003.

Applicants (hereinafter referred to as "Appellants") hereby appeal the rejection of claims 1-14 of the above referenced application.

REAL PARTY IN INTEREST

The present application is assigned to International Business Machines Corp., as evidenced by an assignment recorded September 22, 1999 in the U.S. Patent and Trademark Office at Reel 10279, Frame 0510. The assignee, International Business Machines Corp., is the real party in interest.

RELATED APPEALS AND INTERFERENCES

There are no known related appeals and interferences.

STATUS OF CLAIMS

Claims 1-14 are pending in the present application. Claims 1-14 stand rejected under 35 U.S.C. §103(a). Claims 1-14 are appealed.

STATUS OF AMENDMENTS

There have been no amendments filed subsequent to the rejection.

SUMMARY OF INVENTION

The present invention relates generally to processing of work items in data processing systems and more particularly to the scheduling of tasks to process such work items (Specification, page 1, lines 5-8).

By way of example, as recited in claim 1, a method for processing work items in a data processing system comprises the following steps. An interrupt is generated in response to receipt of a work item in the system. The generated interrupt is serviced to schedule a task for later processing of the work item, without reenabling the interrupt. The task is subsequently executed to process the work item. Finally, a further task is speculatively scheduled for processing of any work items that are subsequently received in the system.

As a further example, as recited in claim 2, a method for processing work items in a data processing system comprises the following steps. The speculatively scheduled task is executed to process any work items received by the system. On a determination that there are no work items to process, the interrupt is enabled. On a determination that there are work items to process, a further task is speculatively scheduled without re-enabling the interrupt.

In accordance with one embodiment of the invention, the method could include the step of continually scheduling speculative tasks (i.e. polling) for processing of work items that may subsequently be received in the system (Specification, page 4, lines 8-16). In a preferred method, when the speculatively scheduled task is executed to process any work items received by the system

and it is determined that there are no work items, the interrupt is enabled. Thus, when the system is fully utilized, the interrupt mechanism is replaced with a polling mechanism involving a continuous series of speculatively scheduled tasks. However, when the system or device utilization decreases (i.e., when there are no work items when the speculatively scheduled task is processed), then the system reverts to interrupts (Specification, page 4, lines 18-27).

Finally, as recited in claim 12, a method for processing work items in a data processing system, comprises the following steps. An interrupt-based mechanism is effectively provided for processing work items when system utilization is low with respect to work items. A polling-based mechanism is effectively provided for processing work items when system utilization is relatively high with respect to work items.

A flow diagram showing the steps involved in a method according to a preferred embodiment of the invention is shown in FIG. 2. Schematic representations of the state of the task and work item queues of the preferred embodiment of the invention at different states of the method of FIG.2 are shown in FIGS. 3A, 3B and 3C. The above-mentioned figures describe a process, operable on a storage controller, for processing work items from the host system in a manner that combines the best attributes of the polling and interrupt methods to service the host work items with low latency at low utilization and by polling for low overhead at high utilization (Specification, page 10, lines 2-8).

ISSUE PRESENTED FOR REVIEW

Whether claims 1-14 are properly rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 4,980,820 to Youngblood (hereinafter "Youngblood").

GROUPING OF CLAIMS

Claims 1-14 do not stand or fall together. More particularly, claims 1, 3-5 and 7-10 stand or fall together, claims 2, 6 and 11 stand or fall together, and claims 12-14 stand or fall together.

ARGUMENT

Appellants incorporate by reference herein the disclosure of all previous responses filed in the present application, namely, responses dated January 24, 2003, July 3, 2003 and September 8, 2003.

With regard to the issue of whether claims 1-14 are properly rejected under 35 U.S.C. 103(a) as being unpatentable over Youngblood, Appellants assert that such claims are patentable for at least the reasons that independent claims 1, 5, 10 and 12 from which claims 2-4, 6-9, 11, 13 and 14 directly depend, are patentable.

A proper 35 U.S.C. §103 rejection must expressly specify how the cited reference teaches or suggests all the claim limitations. Youngblood fails to disclose the individual elements of claim 1.

The present invention, as recited in independent claim 1, recites a method of processing work items in a data processing system, comprising the steps of: (i) generating an interrupt in response to receipt of a work item in the system; (ii) servicing the generated interrupt to schedule a task for later processing the work item, without re-enabling the interrupt; (iii) subsequently executing the task to process the work item; and (iv) speculatively scheduling a further task for processing of any work items that are subsequently received in the system. Independent claims 5 and 10 recite other aspects of the invention comprising similar limitations.

The Office Action states that the difference between Youngblood and the present invention is that Youngblood does not teach servicing the generated interrupt to schedule a task for later processing of the work item, without re-enabling the interrupt. The Office Action then states that it would have been obvious to disable interrupts, since it is defined in Youngblood that the interrupt requests are to be held off for a minimum amount of time in order to service an interrupt request of equal or higher priority. However, Youngblood does not state that interrupt requests are disabled. Claim 1 of the present invention recites the generation of an interrupt and the scheduling of tasks for processing the work item of the interrupt, while keeping interrupts disabled in all cases. Thus, Appellants assert that, it is not obvious to schedule a task for later processing of a work item in servicing a generated interrupt, without re-enabling the interrupt.

Further, Youngblood fails to disclose speculatively scheduling a further task for processing of any work items that are subsequently received in the system. The Office Action refers to column 7, lines 19-52 and column 8, lines 18-26, however Youngblood only teaches the checking of a work queue in a predetermined priority sequence and the performance of items according to detected work queue bits. Claim 1 of the present invention recites a method in which after receiving an interrupt, disabling interrupts, and scheduling tasks for the work item of the original interrupt, a further task is speculatively scheduled for processing any work items subsequently received in the system. As defined by the specification, a task is speculative in nature when the processor is anticipating that further work items will appear on the work item queue while tasks for an original interrupt are executed. Since interrupts are not enabled when the speculative task is scheduled, work items which are subsequently added to the work item queue do not generate work item interrupts, see, e.g., page 12, lines 11-18 of the specification. Therefore, when the speculative task reaches the head of the task queue, the processor may have work items in the work item queue to process. The method of speculatively scheduling tasks is defined in the specification of the present invention as "polling", see, e.g., page 4, lines 8-16 of the specification. Thus, an original interrupt may be followed thereafter by a polling method.

Youngblood does not disclose a system that utilizes a polling method. Youngblood also does not disclose the interaction of a work item queue and a task queue while interrupts are disabled in the manner of the present invention. Finally, Youngblood does not disclose a method that adds a task, speculative in nature, that will reach the head of the task queue and check for work items to process, thus potentially beginning a polling process.

Furthermore, the Federal Circuit has stated that when patentability turns on the question of obviousness, the obviousness determination "must be based on objective evidence of record" and that "this precedent has been reinforced in myriad decisions, and cannot be dispensed with." In re Lee, 277 F.3d 1338, 1343 (Fed. Cir. 2002). Moreover, the Federal Circuit has stated that "conclusory statements" by an examiner fail to adequately address the factual question of motivation, which is material to patentability and cannot be resolved "on subjective belief and unknown authority." Id. at 1343-1344.

In the Office Action at page 3, the Examiner provides the following statements to prove motivation to modify Youngblood, with emphasis supplied: "It would have been obvious . . . to disable interrupts, since it is defined in Youngblood that the interrupt requests are to be held off (disabled) for a minimum amount of time in order to service an interrupt request of equal or higher priority."

Appellants submit that these statements are based on the type of "subjective belief and unknown authority" that the Federal Circuit has indicated provides insufficient support for an obviousness rejection. More specifically, the Examiner fails to identify any objective evidence of record which supports the proposed modification.

Additionally, Appellants assert that dependent claim 2 of the present invention is patentable with respect to Youngblood. Claim 2 recites a method of processing work items in a data processing system, comprising: (i) executing the speculatively scheduled task to process any work items received by the system; (ii) on a determination that there are no work items to be processed, enabling the interrupt; and (iii) on a determination that there are work items to process, speculatively scheduling a further task, without re-enabling the interrupt. Dependent claims 6 and 11 recite other aspects of the invention comprising similar limitations. Youngblood fails to disclose the individual elements of claims 2, 6 and 11.

The Office Action states that the difference between the reference and claim 2 of the present invention is that Youngblood does not teach speculatively scheduling a further task without re-enabling the interrupt. The Office Action then states that it would have been obvious to one of ordinary skill in the art at the time the invention was made to disable interrupts, since it is defined in Youngblood that the interrupt requests are to be held off for a minimum amount of time in order to service an interrupt request of equal or higher priority. However, Youngblood does not state that interrupt requests are disabled. Claim 2 of the present invention recites the enabling of interrupts when there are no work items to be processed, and the continued disabling of interrupts when there are work items to process. Thus, Appellants assert that, it is not obvious to speculatively schedule a further task without re-enabling the interrupt, when there are work items to process.

Youngblood does not disclose the execution of a speculatively scheduled task. Additionally, Youngblood does not disclose that the execution of the speculatively scheduled task results in the

processing of any work items received by the system. Further, Youngblood does not disclose that an interrupt is re-enabled if there are no work items to process, or that a further task is speculatively scheduled without re-enabling the interrupt, if there are work items to process. Claim 2 of the present invention recites the steps taken to determine whether an interrupt-based mechanism or a polling-based mechanism is to be utilized. The option of an interrupt-based mechanism or a polling-based mechanism is not presented in Youngblood. Additionally, the method of deciding which mechanism to use, through speculatively scheduling a task, is also not disclosed in Youngblood.

Furthermore, in the Office Action at page 4, the Examiner provides the following statements to prove motivation to modify Youngblood, with emphasis supplied: "It would have been obvious . . . to disable interrupts, since it is defined in Youngblood that the interrupt requests are to be held of f (disabled) for a minimum amount of time in order to service an interrupt request of equal or higher priority."

Appellants again submit that these statements are based on the type of "subjective belief and unknown authority" that the Federal Circuit has indicated provides insufficient support for an obviousness rejection. More specifically, the Examiner again fails to identify any objective evidence of record which supports the proposed modification.

Finally, Appellants assert that independent claim 12 of the present invention is patentable with respect to Youngblood. Claim 12 recites a method of processing work items where an interrupt-based mechanism for processing work items is provided when system utilization is low with respect to work items. A polling-based mechanism for processing work items is provided when system utilization is relatively high with respect to work items. Youngblood fails to disclose the elements of independent claim 12.

The Office Action states that it would have been obvious to implement the interrupt-based mechanism with few work items queued and the polling mechanism with many work items queued. The present invention differs from Youngblood in that it discloses the use of an interrupt based-mechanism and a polling-based mechanism. Youngblood discloses an interrupt based-mechanism, however, it fails to disclose a polling-based mechanism. The present invention defines "polling" as the continuous scheduling of speculative tasks for processing work items that may subsequently be received in the system (Specification, page 4, lines 9-11). If the system of Youngblood receives a

work item, an interrupt-based mechanism, not a polling-based mechanism, is activated, by enabling the pending interrupt. Therefore, in Youngblood, the pattern of interrupts is not altered. Further, Youngblood does not disclose mechanisms that are dependent on system utilization with respect to work items as recited in claim 12 of the present invention.

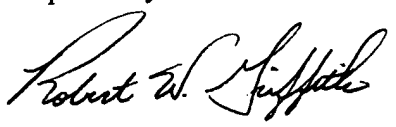
Furthermore, in the Office Action at page 6, the Examiner provides the following statements to prove motivation to modify Youngblood, with emphasis supplied:

> It would have been obvious . . . to implement the interrupt-based mechanism with few work items queued and the polling mechanism with many work items queued. With few items on the queue, the interrupt mechanism would still handle work items. By using the interrupt mechanism there is the potential that another interrupt may be generated and already have a work item relating to that interrupt on that queue, thereby creating an efficient method of handling several work items and satisfying more than one interrupt at a time. On the other hand, if there are a large number of work items on the queue a polling mechanism would be more beneficial, as the polling mechanism can select requests and handle said requests from devices that work items on the queue could satisfy, thereby handling a plurality of interrupts simultaneously. "

Appellants again submit that these statements are based on the type of "subjective belief and unknown authority" that the Federal Circuit has indicated provides insufficient support for an obviousness rejection. More specifically, the Examiner again fails to identify any objective evidence of record which supports the proposed modification.

For at least the reasons given above, Appellants respectfully request withdrawal of the §103(a) rejection of claims 1-14. As such, the application is asserted to be in condition for allowance, and favorable action is respectfully solicited.

Respectfully submitted,



Date: May 3, 2004

Robert W. Griffith
Attorney for Applicant(s)
Reg. No. 48,956
Ryan, Mason & Lewis, LLP
90 Forest Avenue
Locust Valley, NY 11560
(516) 759-4547

APPENDIX

1. A method of processing work items in a data processing system comprising:

generating an interrupt in response to receipt of a work item in the system;

servicing the generated interrupt to schedule a task for later processing of the work item, without re-enabling the interrupt;

subsequently executing the task to process the work item; and

speculatively scheduling a further task for processing of any work items that are subsequently received in the system.

2. A method as claimed in claim 1 comprising the further steps of:

executing the speculatively scheduled task to process any work items received by the system;

on a determination that there are no work items to be processed, enabling the interrupt; and

on a determination that there are work items to process, speculatively scheduling a further task, without re-enabling the interrupt.

3. A method as claimed in claim 1 wherein the work items are managed on a queue.

4. A method as claimed in claim 1 where in the event that further work items are received after the task is scheduled and prior to execution of the task, the step of executing the task comprises processing all the received work items.

5. A data processing system comprising:

processing means for executing tasks to process work items in the data processing system; and interrupt generating means for generating an interrupt in response to receipt of a work item in the system; wherein the processing means is operable to:

service the generated interrupt to schedule a task for later processing of the work item, without re-enabling the interrupt;

subsequently execute the task to process the work item; and

speculatively schedule a further task for processing of any work items that are subsequently received in the system.

6. A data processing system as claimed in claim 5, the processing means being operable on a determination that there are work items to be processed to execute the speculatively scheduled task to process the work items and to schedule a further speculative task; and operable on a determination that there are no work items to be processed to enable the interrupt.

7. A data processing system as claimed in claim 5 further including memory for storing the received work items on a queue.

8. A data processing system as claimed in claim 5 where in the event that further work items are received after the task is scheduled and prior to execution of the task, the processing means is operable to execute the task to process all the work items.

9. A data processing system as claimed in claim 5 wherein the interrupt generating means and processing means are embodied in a data storage controller and the work items comprise data transfer requests from an attached host system.

10. A computer program product comprising a computer usable medium having computer readable program code means embodied in the medium for processing work items in a data processing system, the program code means comprising:

code means for causing the data processing system to service a generated work item interrupt to schedule a task for later processing of the work item, without re-enabling the interrupt;

code means for causing the data processing system to subsequently execute the task to process the work item; and

code means for causing the data processing system to speculatively schedule a further task for processing of any work items that are subsequently received in the system.

11. A computer program product as claimed in claim 10, the computer readable program code means further comprising:

code means for causing the data processing system to execute the speculatively scheduled task to process any work items; and

code means for causing the data processing system to enable the interrupt on a determination that there are no work items for processing.

12. A method of processing work items in a data processing system, comprising:

effectively providing an interrupt-based mechanism for processing work items, when system utiliziation is low with respect to work items; and

effectively providing a polling-based mechanism for processing work items, when system utilization is relatively high with respect to work items.

13. A method as claimed in claim 12 wherein work items are received in accordance with at least one device driver associated with a host system.

14. A method as claimed in claim 12 wherein the data processing system comprises a storage controller.